

Combinatorial coloring of 3-colorable graphs

*Ken-ichi Kawarabayashi**

National Institute of Informatics, Tokyo, Japan

k_keniti@nii.ac.jp

Mikkel Thorup

University of Copenhagen and AT&T Labs—Research

mikkel2thorup@gmail.com

Abstract

We consider the problem of coloring a 3-colorable graph in polynomial time using as few colors as possible. We present a combinatorial algorithm getting down to $\tilde{O}(n^{4/11})$ colors. This is the first combinatorial improvement of Blum's $\tilde{O}(n^{3/8})$ bound from FOCS'90. Like Blum's algorithm, our new algorithm composes nicely with recent semi-definite approaches. The current best bound is $O(n^{0.2072})$ colors by Chlamtac from FOCS'07. We now bring it down to $O(n^{0.2038})$ colors.

*Ken-ichi Kawarabayashi's research is partly supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research, by C & C Foundation, by Kayamori Foundation and by Inoue Research Award for Young Scientists.

1 Introduction

If ever you want to illustrate the difference between what we consider hard and easy to someone not from computer science, use the example of 2-coloring versus 3-coloring: suppose there is too much fighting in a class, and you want to split it so that no enemies end up in the same group. First you try with a red and a blue group. Put someone in the red group, and everyone he dislikes in the blue group, everyone they dislike in the red group, and so forth. This is an easy systematic approach. Digging a bit deeper, if something goes wrong, you have an odd cycle, and it is easy to see that if you have a necklace with an odd number of red and blue beads, then the colors cannot alternate perfectly. Knowing that red and blue do not suffice, we might try introducing green, but this is already beyond what we believe computers can do.

Three-coloring is a classic NP-hard problem. It was proved hard by Garey, Johnson, and Stockmeyer at STOC'74 [9], and was the prime example of NP-hardness mentioned by Karp in 1975 [12]. It is an obvious target for any approach to NP-hard problems. With the approximation approach, given a 3-colorable graph, we try to color it in polynomial time using as few colors as possible. This challenge has engaged many researchers. At STOC'82, Wigderson [15] got down to $O(n^{1/2})$ colors for a graph with n vertices. Berger and Rompel [3] improved this to $O((n/(\log n))^{1/2})$. Blum [4] came with the first polynomial improvements, first to $\tilde{O}(n^{2/5})$ colors at STOC'89, and then to $\tilde{O}(n^{3/8})$ colors at FOCS'90.

The next big step at FOCS'94 was by Karger, Motwani, Sudan [11] who used semi-definite programming (SDP). For a graph with maximal degree Δ_{\max} , they got down to $O(\Delta_{\max}^{1/3})$ colors. Combining it with Wigderson's algorithm, they got down to $O(n^{1/4})$ colors. Later Blum and Karger [5] combined the SDP [11] with Blum's [4] algorithm, yielding an improved bound of $\tilde{O}(n^{3/14}) = \tilde{O}(n^{0.2142})$ (actually we can get $n^{0.2142} + 2$ colors since we have rounded the exponent up). Later improvements in semi-definite programming have also been combined with Blum's algorithm. At STOC'06, Arora, Chlamtac, and Charikar [1] got down to $O(n^{0.2111})$ colors. The proof in [1] is based on the seminal result of Arora, Rao and Vazirani [2] which gives an $O(\sqrt{\log n})$ algorithm for the sparsest cut problem. The last improvement was at FOCS'07 by Chlamtac [6] who got down to $O(n^{0.2072})$ colors.

Only a few lower bounds are known. The strongest known hardness result shows that it is NP-hard to get down to 5 colors [10, 13]. Recently, Dinur, Mossel and Regev [7] showed that it's hard to color with any constant number of colors (i.e., $O(1)$ colors) based on a variant of Unique Games Conjecture.

Some integrality gap results [8, 11, 14] show that the simple SDP relaxation has integrality gap at least $n^{0.157}$. It is therefore natural to go back and see if we can improve things combinatorially.

In this paper, we present the first improvement on the combinatorial side since Blum in 1990 [4]. With a purely combinatorial approach, we get down to $\tilde{O}(n^{4/11})$ colors. Combining it with Chlamtac's SDP [6], we get down to $O(n^{0.2038})$ colors..

Techniques In the details we reuse a lot of the techniques pioneered by Blum [4], but our overall strategy is more structural. We will be looking for sparse cuts that we can recurse over. When no more sparse cuts can be found (and if we are not done by other means), we will have crystallized a vertex set X that is guaranteed to be monochromatic in some 3-coloring. The vertices in X can then be identified. We note for comparison that one of the main technical lemmas in Blum [4] (see Lemma 1 below) is a test to see if a sufficiently large set Y is multichromatic in the sense that no color can dominate too much in any coloring. The monochromatic set X that we identify is typically much too small for Blum’s lemma to apply.

Below we focus on our combinatorial algorithm. The integration with SDP is essentially explained in [5] and is sketched in the end.

2 Preliminaries including ingredients from Blum

We are given a 3-colorable graph $G = (V, E)$ with $n = |V|$ vertices. For a vertex set $X \subseteq V$, let $N(X)$ be the neighborhood of X and $G|X$ be the subgraph induced by X . The unknown 3-coloring is with red, green, and blue. If we say a vertex set X is *green*, we mean that every vertex in X is colored by green.

For some color target k which is polynomial in n , we wish to find a $\tilde{O}(k)$ coloring of G in polynomial time. We are going to reuse several ideas and techniques from Blum’s approach [4].

Progress Blum has a general notion of *progress towards $\tilde{O}(k)$ coloring* (or *progress* for short if k is understood) which is defined such that if we for any 3-colorable graph can make progress towards $\tilde{O}(k)$ coloring in polynomial time, then we can $\tilde{O}(k)$ color any 3-colorable graph in polynomial time.

Blum defines several types of progress, but here the only concrete type of progress we need to know is that of *monochromatic progress* where we identify a set of vertices that is monochromatic in some 3-coloring, hence which can be identified in a single vertex. Otherwise we will only make progress via results of Blum presented below using a common parameter

$$\Psi = n/k^2. \tag{1}$$

A very useful tool we get from Blum is the following multichromatic test:

Lemma 1 ([4, Corollary 4]) *Given a vertex set $X \subseteq V$ of size at least $\Psi = n/k^2$, in polynomial time, we can either make progress towards $\tilde{O}(k)$ -coloring of G , or else guarantee that under every legal 3-coloring of G , the set X is multichromatic.*

In fact Blum has a stronger lemma [4, Lemma 12] guaranteeing not only that X is multichromatic, but that no single color is used by more than a fraction $(1 - 1/(4 \log n))$ of the vertices in X . This stronger version is not needed here. Using Lemma 1 he proves:

Lemma 2 ([4, Theorem 3]) *If two vertices have more than Ψ common neighbors, we can make progress towards $O(k)$ coloring. Hence we can assume that no two vertices have more than Ψ common neighbors.*

Using this bound on joint neighborhoods, Blum proves the following lemma (which he never states in this general quotable form):

Lemma 3 *If the vertices in a set Z on the average have d neighbors in U , then the whole set Z has at least $\min\{d/\Psi, |Z|\} d/2$ distinct neighbors in U .*

Proof If $d/\Psi \leq 2$, the result is trivial, so $d/\Psi \geq 2$. It suffices to prove the lemma for $|Z| \leq d/\Psi$, for if Z is larger, we restrict our attention to the d/Ψ vertices with most neighbors in U . Let the vertices in Z be ordered by decreasing degree into U . Let d_i be the degree of vertex v_i into U . We now study how the neighborhood of Z in U grows as we include the vertices v_i . When we add v_i , we know from Lemma 2 that its joint neighborhood with any (previous) vertex v_h , $h < i$ is of size at most Ψ . It follows that v_i adds at least $d_i - (i-1)\Psi$ new neighbors in U , so $|N(Z) \cap U| \geq \sum_{i=0}^{|Z|-1} (d_i - (i-1)\Psi) > |Z|d/2$. ■

Second neighborhood structure Let Δ_{\min} be the smallest degree in graph G . With color target k , we can trivially assume $\Delta_{\min} > k$ (but Δ_{\min} may be much larger if we apply SDP to low degree vertices). For some $\Delta_0 = \tilde{\Omega}(\Delta_{\min})$, Blum [4, Theorems 7 and 8 and the Proof of Theorem 5] identifies in polynomial time a subgraph H_0 of G that consists of a vertex r_0 and two disjoint vertex sets S_0, T_0 , with the following 3-level structure:

- A root vertex r_0 . We assume r_0 is colored red in any 3-coloring.
- A first neighborhood $S_0 \subseteq N_G(r_0)$ of size at least Δ_0 .
- A second neighborhood $T_0 \subseteq N_G(S_0)$ of size at most n/k .
- All edges in H_0 go between r_0 and S_0 and between S_0 and T_0 .
- The vertices in S_0 have average degree Δ_0 into T_0 .
- The degrees from T_0 to S_0 are all within a factor $(1 \pm o(1))$ around an average $\delta_0 \geq \Delta_0^2 k/n$.

Note that [4, Theorems 7 and 8] does not have this size bound on T_0 . Instead there is a large set R of red vertices leading to a large identifiable independent set constituting a constant fraction of T_0 . If this set is of size $\tilde{\Omega}(n/k)$, then Blum makes progress towards a $\tilde{O}(k)$ coloring as described in [4, Proof of Theorem 5], and we are done. Assuming that this did not happen, we have the size bound on T_0 .

Blum seeks progress directly in the above structure, but we are going to apply a series of reductions which either make progress, find a good cut recursing on one side, or identify a monochromatic set. This is why we already now used the subscript $_0$ to indicate the original structure provided by Blum [4].

3 Our coloring algorithm

We will use Blum's second neighborhood structure H_0 with a color target

$$k = \tilde{\Theta} \left((n/\Delta_{\min})^{4/7} \right). \quad (2)$$

We are going to work on induced subproblems $(S, T) \subseteq (S_0, T_0)$ defined in terms of a subsets $S \subseteq S_0$ and $T \subseteq T_0$. The edges considered in the subproblem are exactly those from H_0 between S and T . This edge set is denoted $E_0(S, T)$. With r_0 red in any 3-coloring, we know that all vertices in $S \subseteq S_0$ are blue or green. We will define *high degrees* in T (to S) as degrees bigger than $\delta_0/16$ (almost a factor 16 below the average in T_0), and we will make sure that any subproblem (S, T) considered satisfies:

- (i) We have more than Ψ vertices of high degree in T .

Cut-or-color We will implement a subroutine $\text{cut-or-color}(t, S, T)$ which for a problem $(S, T) \subseteq (S_0, T_0)$ takes starting point in an arbitrary high degree vertex $t \in T$. It will have one of the following outcomes:

- Reporting a “sparse cut around a subproblem $(S', T') \subseteq (S, T)$ ” with no cut edges between S' and $T \setminus T'$ and only few cut edges between T' and $S \setminus S'$. The exact definition of a sparse cut is complicated, but at this point, all we need to know is that cut-or-color may declare a sparse cut.
- Some progress toward k -coloring with Blum's Lemma 1.
- A guarantee that if r and t have the different colors in some 3-coloring C_t of G , then S is monochromatic in C_t .

We note that testing whether or not S can be monochromatic is only new if $|S| < \Psi$, for if $|S| \geq \Psi$ and S was monochromatic, we would get immediate progress with Lemma 1.

Recurring towards a monochromatic set Using cut-or-color , we now describe our main recursive algorithm which takes as input a subproblem (S, T) . Let U be the set of high degree vertices in T . By (i) we have $|U| \geq \Psi$, so we can apply Blum's multicolor test from Lemma 1. Assuming we did not make progress, we know that U is multichromatic in every valid 3-coloring. We now apply cut-or-color to each $t \in U$, stopping only if a sparse cut is found or progress is made. If we make progress, we are done, so assume that this does not happen.

Monochromatic case The most interesting case is if we get neither progress nor a sparse cut.

Lemma 4 *If cut-or-color does not find progress or a sparse cut for any high degree $t \in U$, then S is monochromatic in some 3-coloring of G .*

Proof Since U is multichromatic in any 3-coloring, there is a 3-coloring C_t where some $t \in U$ has a different color than r_0 , and then **cut-or-color** guarantees that S is monochromatic in C_t . ■

Thus we have found a monochromatic set, so monochromatic progress can be made.

Sparse cut If a sparse cut around a subproblem (S', T') is reported, we recurse on (S', T') .

4 Implementing cut-or-color

We are now going to implement **cut-or-color**. The first part of it is essentially the coloring that Blum [4, §5.2] uses for dense regions. We shall describe how we bypass the limits of his approach as soon as we have presented his part.

Assume that $t \in U$ and r have different colors in some coloring C_t ; otherwise the algorithm provides no monochromatic guarantee. Let us say that r_0 is red and t is green.

Let X be the neighborhood of t in S and let Y be the neighborhood of X in T . As in [4] we note that all of X must be blue, and that no vertex in Y can be blue. We are going to expand $X \subseteq S$ and $Y \subseteq T$ preserving the following invariant:

- (ii) If t was green and r_0 was red, then X is all blue and Y has no blue.

If we end up with $X = S$, then (ii) implies that S is monochromatic in any 3-coloring where r_0 and t have different colors.

X -extension Now consider any vertex $s \in S$ whose degree into Y is at least Ψ . Using Lemma 1 we can check that $N(s) \cap Y$ is multichromatic. Since Y has no blue, we conclude that s has both red and green neighbors, hence that s is blue. Note conversely that if s was green, then all its neighbors in Y would have to be red, and then the multichromatic test from Lemma 1 would have made progress. Preserving (ii), we now add the blue s to X and all neighbors of s in T to Y . We shall refer to this as an X -extension.

Relation to Blum's algorithm Before continuing, let us briefly relate to Blum's [4] algorithm. The above X -extension is essentially the coloring Blum [4, §5.2] uses for dense regions. He applies it directly to his structure H_0 from Section 2. He needs a larger degree $\delta_0 \geq \Delta_0^2 k/n$ than we do, but then he proves that the set of vertices s with degree at least Ψ into Y is more than Ψ . This means that either he finds progress with a green vertex s , or he ends up with a blue set X of size Ψ , and gets progress applying Lemma 1 to X .

Our algorithm works for a smaller δ_0 and thereby for a smaller color target k . Our extended X is typically too small for Lemma 1. In fact, as we recurse, we will get sets S that themselves are much smaller than Ψ . Otherwise we would be done with Lemma 1 if S was monochromatic.

Below we introduce Y -extensions. They are similar in spirit to X -extensions, and would not help us if we like Blum worked directly with H_0 . The important point will be that if we do not end up with $X = S$, and if neither extension is possible, then we have identified a sparse cut that we can use for recursion. We are thus borrowing from Blum's proof [4] in the technical details, but the overall strategy, seeking sparse cuts for recursion to crystallize a small monochromatic set S , is entirely different (and new).

Y -extension We now describe a Y -extension, which is similar in spirit to the X -extension, but which will cause more trouble in the analysis. Consider a vertex t' from $T \setminus Y$. Let X' be its neighborhood in S (note that $X' \cap X = \emptyset$) and Y' be the neighborhood of X' in T . Suppose $|Y \cap Y'| \geq \Psi$. Using Lemma 1 we check that $Y \cap Y'$ is multichromatic. We claim that t' cannot be blue, for suppose it was. Then its neighborhood would have no blue and S is only blue and green, so X' is all green. Then Y' has no green, but Y has no blue, so $Y' \cap Y$ would be all red, contradicting that $Y \cap Y'$ is multichromatic. We conclude that t' is not blue. Preserving (ii), we now add t' to Y .

Closure We are going to extend X and Y as long as possible. Suppose we end up with $X = S$. With (ii) **cut-or-color** declares that S is monochromatic in any 3-coloring where r and t have different colors.

Otherwise we are in a situation where no X -extension nor Y -extension is possible, and then **cut-or-color** will declare a sparse cut around (X, Y) . A *sparse cut around (X, Y)* is simply defined as being obtained this way. It has the following properties:

- (iii) The original high degree vertex t has all its neighbors from S in X , that is, $N(t) \cap S \subseteq X$.
- (iv) There are no edges between X and $T \setminus Y$. To see this, recall that when an X -extension adds s' to X , it includes all its neighbors in Y . The Y -extension does not change X .
- (v) Each vertex $s' \in S \setminus X$ has $|N(s') \cap Y| < \Psi$.
- (vi) Each vertex $t' \in T \setminus Y$ has $|N(N(t')) \cap Y| < \Psi$.

A most important point here is that this characterization of a sparse cut does not depend on the assumption that t and r have different colors in some 3-coloring. It only assumes that X and Y cannot be extended further.

5 Correctness

It should be noted that the correctness of **cut-or-color** follows from (ii) which is immediate from the construction. The technical difficulty that remains is to ensure that we never end up considering a subproblem with too few high-degree vertices for (i), hence where we cannot apply Lemma 1 to ensure that the high degree vertices do not all have the same color as r_0 .

6 Degree constraints

Before we can start our recursive algorithm, we need some slightly different degree constraints from those provided by Blum [4] described in Section 2:

- The vertices in S_0 have average degree Δ_0 into T_0 .
- The degrees from T_0 to S_0 are all within a factor $(1 \pm o(1))$ around the average $\delta_0 \geq \Delta_0^2 k/n$.

We need some initial degree lower bounds, which are obtained simply by removing low degree vertices creating our first induced subproblem $(S_1, T_1) \subseteq (S_0, T_0)$. Starting from $(S_1, T_1) = (S_0, T_0)$, we repeatedly remove vertices from S_1 with degree to T_1 below $\Delta_0/4$ and vertices from T_1 with degree to S_1 below $\delta_0/4$ until there are no such low-degree vertices left. The process eliminates less than $|S_0|\Delta_0/4 + |T_0|\delta_0/4 = |E_0(S_0, T_0)|/2$ edges, so half the edges of $E_0(S_0, T_0)$ remain in $E_0(S_1, T_1)$. We also note that the average degree in T remains above $\delta_0/2 = 2\delta_1$. The point is that the average on the T -side only goes down when we remove low degree vertices from S_0 , and that can take away at most $1/4$ of the edges. With $\Delta_1 = \Delta_0/4$ and $\delta_1 = \delta_0/4$, we get

- The degrees from S_1 to T_1 are at least Δ_1 .
- The degrees from T_1 to S_1 are between δ_1 and $(1 + o(1))\delta_0 < 5\delta_1$, with an average above $2\delta_1$. Note that $\delta_1 = \delta_0/4 \geq \Delta_0^2 k/(4n) = 4\Delta_1^2 k/n$.

A *high degree* in T can now be restated as a degree to S above $\delta_1/4 = \delta_0/16 = \Delta_1^2 k/n$.

Lemma 5 $\Delta_1 = \Psi n^{\Omega(1)}$.

Proof Recall that $\Delta_1 = \Delta_0/4 = \tilde{\Omega}(\Delta_{\min})$, so from (2) we get $k = \tilde{O}((n/\Delta_1)^{4/7})$, or equivalently, $\Delta_1 = \tilde{\Omega}(n/k^{7/4})$. Since $\Psi = n/k^2$, we conclude that $\Delta_1 = \tilde{\Omega}(\Psi k^{1/4}) = \Psi n^{\Omega(1)}$. ■

Recursion Our recursion will start from $(S, T) = (S_1, T_1) \subseteq (S_0, T_0)$. We will ensure that each of subproblems (S, T) considered satisfies the following degree invariants:

- (vii) Each vertex $v \in S$ has all its neighbors from T_1 in T , so the the degrees from S to T remain at least Δ_1 . This invariant follows immediately from sparse cut condition (iv).
- (viii) The average degree in T to S is at least $\delta_1/2$.

The following key lemma shows that the degree constraints imply that we have enough high degree vertices in T that we can test if they are multichromatic with Lemma 1.

Lemma 6 (vii) and (viii) imply (i), i.e., that we have more than Ψ vertices of high degree in T .

Proof If h is the fraction of high degree vertices in T , the average degree in T is at most $h \cdot 5\delta_1 + (1-h)\delta_1/4$ which by (viii) is at least $\delta_1/2$. Hence $h = \Omega(1)$. By (vii) we have $|T| \geq \Delta_1$, so we have $\Omega(\Delta_1)$ high degree vertices in T . By Lemma 5 this is much more than Ψ high degree vertices. ■

7 Maintaining degrees recursively

All that remains is to prove that invariant (viii) is preserved, i.e., that the average degree from T to S does not drop below half the original minimum degree δ_1 from T_1 to S_1 . Inductively, when a sparse cut is declared around a new subproblem $(S', T') = (X, Y) \subseteq (S, T)$, we can assume that (vii) and (viii) are satisfied for (S, T) and that (vii) is satisfied for (X, Y) . It remains to prove (viii) for (X, Y) .

Below we first show that when a sparse cut is declared around $(X, Y) \subseteq (S, T)$, then X cannot be too small. We later complement this by showing that the total number of edges cut in the recursion cannot be too large.

Lemma 7 $|Y| \geq \Delta_1^2 k^2 / (8n)$.

Proof If t is the high degree vertex we started with in T , then by (iii), we have the whole neighborhood Z of t in S preserved in X . By definition of high degree, $|Z| \geq \delta_1/4 = \Delta_1^2 k/n$. By (vii) the degrees from Z to Y are at least $d = \Delta_1/2$, so $d/\Psi = \Delta_1/2 \cdot k^2/n = o(|Z|)$. Hence by Lemma 3, we have $|N(Z) \cap Y| \geq d/\Psi \cdot d/2 = \Delta_1^2/(8\Psi) = \Delta_1^2 k^2/(8n)$. ■

In our original problem (S_1, T_1) , each vertex $v \in Y$ we had δ_1 edges to S_1 , so the original number of edges from Y to S_1 was at least

$$\delta_1 \Delta_1^2 k^2 / (8n). \quad (3)$$

To prove (viii), we argue that at least half of these edges are between Y and X . This follows if we can prove that the total number of edges cut is only half the number in (3).

The following main technical lemma relates the number of new cut edges around the subproblem (X, Y) to the reduction $|T \setminus Y|$ in the size of the T -side:

Lemma 8 *The number of cut edges from Y to $S \setminus X$ is bounded by*

$$|T \setminus Y| \frac{40\delta_1 n^2}{\Delta_1^2 k^4}. \quad (4)$$

Proof First we note that from (v) we get a trivial bound of $\Psi|S \setminus X|$ on the number of new cut edges, but is not strong enough for (4). Here we use (vi) we get

$$\sum_{y \in Y} |N(N(y)) \setminus Y| = \sum_{t' \in T \setminus Y} |N(N(t')) \cap Y| \leq |T \setminus Y| \Psi = |T \setminus Y| n/k^2. \quad (5)$$

We will now, for any $y \in Y$, relate $|N(N(y)) \setminus Y|$ to the number $|N(y) \setminus X|$ of edges cut from y to $S \setminus X$. Let $Z = N(y) \setminus X$. By (iv) we have that $N(N(y)) \setminus Y = N(Z) \setminus Y$. Consider any vertex $v \in Z$. By (vii) the degree from v to T is at least Δ_1 . Since $v \notin X$, by (v), the degree from v to Y is at most Ψ , and by Lemma 5, $\Psi = o(\Delta_1)$. The degree from v to $T \setminus Y$ is therefore at least $(1 - o(1))\Delta_1 \geq \Delta_1/2$. This holds for every $v \in Z$. It follows by Lemma 3 that $|N(Z) \setminus Y| \geq \min\{(\Delta_1/2)/\Psi, |Z|\} \Delta_1/4$. Relative to $|Z|$, this is

$$\frac{|N(Z) \setminus Y|}{|Z|} \geq \min \left\{ \frac{\Delta_1/(2\Psi)}{|Z|}, 1 \right\} \Delta_1/4$$

From our original configuration (S_1, T_1) , we know that all degrees in T are bounded by $5\delta_1$ and this bounds the size of $Z = N(y)$. Therefore

$$\frac{\Delta_1/(2\Psi)}{|Z|} \geq \frac{\Delta_1 k^2/(2n)}{5\delta_1} = \frac{\Delta_1 k^2}{10\delta_1 n}.$$

Since $\delta_1 \geq 4\Delta_1^2 k/n$, we have

$$\frac{\Delta_1 k^2}{10\delta_1 n} \leq \frac{\Delta_1 k^2}{10(4\Delta_1^2 k/n)n} = \frac{k}{40\Delta_1} < 1.$$

Therefore

$$\frac{|N(Z) \setminus Y|}{|Z|} \geq \min \left\{ \frac{\Delta_1/(2\Psi)}{|Z|}, 1 \right\} \Delta_1/4 \geq \frac{\Delta_1 k^2}{10\delta_1 n} \Delta_1/4 = \frac{\Delta_1^2 k^2}{40\delta_1 n}.$$

Recalling $Z = N(y) \setminus X$ and $N(Z) \setminus Y = N(N(y)) \setminus Y$, we rewrite the inequality as

$$|N(y) \setminus X| \leq \frac{40\delta_1 n}{\Delta_1^2 k^2} |N(N(y)) \setminus Y|.$$

Using (5), we now get the desired bound on the number of cut edges from Y to $S \setminus X$:

$$\sum_{y \in Y} |N(y) \setminus X| \leq \frac{40\delta_1 n}{\Delta_1^2 k^2} \sum_{y \in Y} |N(N(y)) \setminus Y| = \frac{40\delta_1 n}{\Delta_1^2 k^2} |T \setminus Y| n/k^2 = |T \setminus Y| \frac{40\delta_1 n^2}{\Delta_1^2 k^4}.$$

■

From Lemma 8 it immediately follows that the total number of edges cut in the whole recursion is at most

$$|T_1| \frac{40\delta_1 n^2}{\Delta_1^2 k^4} \leq \frac{40\delta_1 n^3}{\Delta_1^2 k^5}. \quad (6)$$

This should be at most half the original number of edges from (3). Thus we maintain (vii) with an average degree of $\delta_1/2$ from Y as long as

$$\frac{40\delta_1 n^3}{\Delta_1^2 k^5} \leq \frac{\delta_1 \Delta_1^2 k^2}{16n}$$

or equivalently

$$k^7 \geq 640 (n/\Delta_1)^4. \quad (7)$$

Thus, if k satisfies (7), then all our degree constraints are maintained, which means that we will keep recursing over sparse cuts until we either make progress towards a $\tilde{O}(k)$ coloring, or end up with a provably monochromatic set S on which we can make monochromatic progress towards $\tilde{O}(k)$ coloring. Since $\Delta_1 = \Omega(\Delta_0) = \tilde{\Omega}(\Delta_{\min})$, we can pick k as a function of Δ_{\min} and n such that $k = \tilde{O}((n/\Delta_{\min})^{4/7})$ and such that (7) will be satisfied. Thus we conclude

Theorem 9 *If a 3-colorable graph has minimum degree Δ_{\min} , then we can make progress towards $\tilde{O}((n/\Delta_{\min})^{4/7})$ coloring in polynomial time.*

Since we make trivial progress for vertices of degree below k , we can assume $\Delta_{\min} \geq k$, and hence we can balance with $k = \tilde{\Theta}(n^{4/11})$ for a purely combinatorial algorithm to obtain the following corollary.

Corollary 10 *A 3-colorable graph on n vertices can be colored with $\tilde{O}(n^{4/11})$ colors in polynomial time.*

8 Integration with SDP

We will now show how to combine our combinatorial algorithm with the best SDP coloring of Chlamtac.

Lemma 11 ([6, Corollary 16]) *For any 3-colorable graph G on n vertices with maximal degree $\Delta_{\max} \leq n^{0.6546}$, in polynomial time, we can find an independent set of size $\Omega(n/\Delta_{\max}^{0.3166})$. Hence we can make progress towards $O(\Delta_{\max}^{0.3166})$ coloring.*

The above statement fixes a small typo in the statement of [6, Corollary 16] which says that the maximal degree should be below $\Delta_{\max} = n^{0.6546}$, as if we wouldn't benefit from a smaller Δ_{\max} .

We note that combination with Theorem 9 would be trivial if the Δ_{\min} in Theorem 9 was equal to the Δ_{\max} in Lemma 11. Things are not that simple, but Karger and Blum [5] have already shown how to combine the original SDP of Karger et al. [11] and Blum's algorithm [4].

To describe the combination in our case, we first need to elaborate a bit on Blum's progress from Section 2. We already mentioned monochromatic progress where we identify a set of vertices that we know are monochromatic in some coloring.

Less immediate types of progress work as follows. We start with a graph $G = G_0$. We have a constant number of players $i = 1, \dots, \ell = O(1)$ that can announce progress towards $\tilde{O}(k)$ coloring. Each player i

starts with an empty vertex set V_i . When player i announces progress, he removes some vertices from G and place them in his set V_i . His promise is that if he ends up with $n_i = |V_i| \geq n/(2\ell) = \Omega(n)$ vertices, then he can find an independent set I_i of size $n_i/\tilde{O}(k)$ in the subgraph $G_0|V_i$ of G_0 induced by V_i .

The players play until G has lost half its vertices. This means that we only play on G when G has at least $n/2$ vertices left. When they play, we always need someone to claim progress. Combined the players end up taking more than $n/2$ vertices, so some player i ends up with $n_i \geq n/(2\ell) = \Omega(n)$ vertices. His independent set I_i is of size $n_i/\tilde{O}(k) = n/\tilde{O}(k)$, and I_i is also independent in G_0 . We can therefore use a single color on I_i and recurse on $G_0 \setminus I_i$.

The formal proof that the game leads to a $\tilde{O}(k)$ coloring, including the special monochromatic player, is described in [4].

We now introduce an SDP player s that for some desired Δ claims progress if there is any vertex v of degree below Δ , and moves v to his set V_s . The SDP player will end up with an induced subgraph $G_s = G|V_s$ of G where the average degree is below 2Δ . We delete all vertices with degree at least 4Δ . The resulting vertex set V'_s has $|V'_s| > n_s/2$ and $G'_s = G|V'_s$ has maximum degree $\Delta_{\max} < 4\Delta$. If $n_s = |V_s| = \Omega(n)$ and $\Delta = o(n^{0.6546})$, he can apply Lemma 11 to G'_s and get an independent set of size $\Omega(n'_s/\Delta_{\max}^{0.3166}) = \Omega(n/\Delta^{0.3166})$. The SDP player thus follows the rule for progress towards $O(\Delta^{0.3166})$ coloring.

Since it for any graph suffices that some player can make progress, a player like our combinatorial algorithm in Theorem 9 can wait for the SDP player to remove all the vertices of degree at most Δ . We therefore only apply Theorem 9 to graphs with minimum degree $\Delta_{\min} \geq \Delta$. With Theorem 9, we then make progress towards $\tilde{O}((n/\Delta_{\min})^{4/7}) = \tilde{O}((n/\Delta)^{4/7})$ coloring in polynomial time.

Balancing $(n/\Delta)^{4/7} = \Delta^{0.3166}$, we get $\Delta = n^{0.6435} = o(n^{0.6546})$, and conclude

Theorem 12 *For any 3-colorable graph G , there is a polynomial time algorithm to make progress towards an $O(n^{0.2038})$ -coloring.*

Hence we get

Corollary 13 *A 3-colorable graph on n vertices can be colored with $O(n^{0.2038})$ colors in polynomial time.*

References

- [1] S. Arora, E. Chlamtac, and M. Charikar. New approximation guarantee for chromatic number. In *Proc. 38th STOC*, pages 215–224, 2006.
- [2] S. Arora, S. Rao, and U. Vazirani. Expanders, geometric embeddings and graph partitioning. *J. ACM*, 56(2):1–37, 2009. Announced at STOC'04.

- [3] B. Berger and J. Rompel. A better performance guarantee for approximate graph coloring. *Algorithmica*, 5(3):459–466, 1990.
- [4] A. Blum. New approximation algorithms for graph coloring. *J. ACM*, 41(3):470–516, 1994. Announced at STOC’89 and FOCS’90.
- [5] A. Blum and D. Karger. An $\tilde{O}(n^{3/14})$ -coloring algorithm for 3-colorable graphs. *Inf. Process. Lett.*, 61(1):49–53, 1997.
- [6] E. Chlamtac. Approximation algorithms using hierarchies of semidefinite programming relaxations. In *Proc. 48th FOCS*, pages 691–701, 2007.
- [7] I. Dinur, E. Mossel, and O. Regev. Conditional hardness for approximate coloring. *SIAM J. Comput.*, 39(3):843–873, 2009. Announced at STOC’06.
- [8] U. Feige, M. Langberg, and G. Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. In *Proc. 43rd FOCS*, pages 283–292, 2002.
- [9] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. Announced at STOC’74.
- [10] V. Guruswami and S. Khanna. On the hardness of 4-coloring a 3-colorable graph. *SIAM Journal on Discrete Mathematics*, 18(1):30–40, 2004.
- [11] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998. Announced at FOCS’94.
- [12] R. M. Karp. On the computational complexity of combinatorial problems. *Networks*, 5:45–68, 1975.
- [13] S. Khanna, N. Linial, and S. Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000.
- [14] M. Szegedy. A note on the θ number of Lovász and the generalized Delsarte bound. In *Proc. 35 FOCS*, pages 36–39, 1994.
- [15] A. Wigderson. Improving the performance guarantee for approximate graph coloring. *J. ACM*, 30(4):729–735, 1983. Announced at STOC’82.